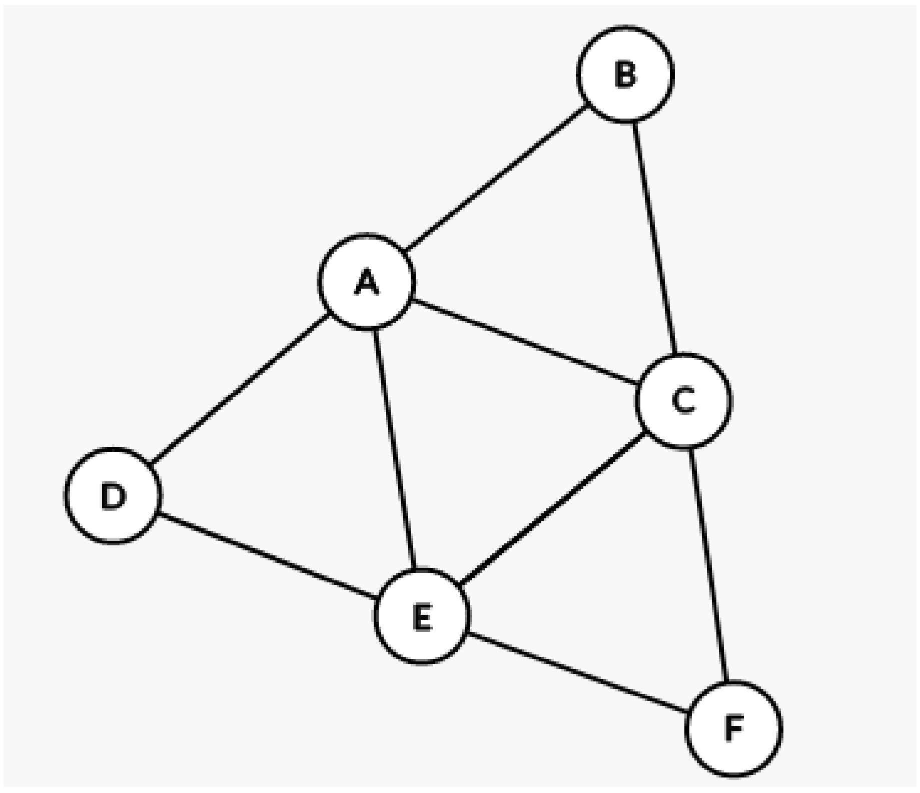
# 4. Fai

**Program Name: Fai.java** **Input File: fai.dat**

Fai just got a job working for the UIL's virtual reality team. In an attempt to get all K-12 students more familiar with their assigned district, the UIL wants to make a warped speed video showing what each student would see on a bus ride to another school's city or town. This way, students will have a general idea as to what they will see out of their school bus window before ever making the trek to the other campus.

To accomplish this, the UIL virtual reality team has provided Fai with a 360-degree camera attached to the top of her car. Fai knows she needs to drive every single road between two cities exactly one time to record the street view, so that all possible routes have been recorded. With the rising price of fuel, and in an attempt to be as efficient as Fai can be with her time, Fai knows she needs to start recording and end recording in the same city, if possible. With the task of driving every single road, Fai knows she may visit the same city or town multiple times, which is okay, she just has to travel all the given roads the UIL assigns to her to drive.

For example, say that City A, City B, City, C, City D, City E, and City F are all in the same district and the following roads exist: City A<->City B,City A<->City C,City A<->City E,City A<->City D,City B<->City C,
City C<->City F,City C<->City E,City F<->City E,City E<->City D. If Fai were to start in City A she could do the following road traversal to cover all roads exactly once, starting and stopping from City A:



City A -> City D -> City E -> City F -> City C -> City B -> City A -> City E -> City C -> City A

Can you help Fai write a program that, given a district and all the roads between cities or towns, can determine if it's possible to drive all roads starting and stopping from a given city or town?

***Continued next page…***

***Fai, continued***

**Input:** The input will consist of an integer *T*, the number of test cases. *T* will be in the range of [1,10]. For each test case, input will consist of four lines. Line 1 will contain the name of all the cities or towns in the district. The number of cities or towns will be greater than or equal to two, and will not exceed 10.. Cities or towns are not limited to one word names. It will be guaranteed that no two cities or towns have the same exact name. Names in the list will be separated by a comma ",". Line 2 contains the name of the start city or town where Fai will begin recording street views. Line 3 will consist of the roads Fai has been assigned to record by the UIL. The roads will be given in the form: "`Name1<->Name2`" this means a road exists between Name1 and Name2 and must be driven by Fai exactly once. A road in this problem is always two-way. Roads will be separated by a comma ",". There will be at least one road present and there will always exist a path, or series of road(s), between two cities or towns. Line 4 is 20 dashes and serves to separate all test cases.

**Output:** For each test case, you are to output "Test Case #: possible" if Fai can start and stop at the given start city, traversing each road exactly once or "Test Case#: impossible" if Fai cannot start and stop at the given start city, traversing each road exactly once.

**Sample input:**
```
9
City A,City B,City C,City D,City E,City F
City A
City A<->City B,City A<->City C,City A<->City E,City A<->City D,City B<->City C,City C<->City F,City
C<->City E,City F<->City E,City E<->City D
--------------------
Town 0,Town 1,Town 2,Town 3,Town 4
Town 3
Town 1<->Town 2,Town 1<->Town 0,Town 2<->Town 0,Town 0<->Town 4,Town 0<->Town 3,Town 3<->Town 4
--------------------
Dallas,San Antonio,Houston,El Paso,Austin
Houston
Dallas<->San Antonio,Dallas<->Houston,Dallas<->El Paso,Dallas<->Austin,San Antonio<->Houston,San
Antonio<->El Paso,San Antonio<->Austin,El Paso<->Austin
--------------------
Wall,College Station,San Angelo
Wall
Wall<->College Station,College Station<->San Angelo,San Angelo<->Wall
--------------------
Lubbock,Loredo,Lampasas,Liberty Hill,Lago Vista
Lubbock
Lubbock<->Loredo,Lubbock<->Lampasas,Lubbock<->Liberty Hill,Lubbock<->Lago Vista
--------------------
Abilene,Beaumont,Childress,Dalhart,Eden,Fort Worth
Beaumont
Abilene<->Fort Worth,Abilene<->Eden,Abilene<->Childress,Abilene<->Beaumont,Beaumont<-
>Childress,Childress<->Dalhart,Childress<->Eden,Dalhart<->Eden,Eden<->Fort Worth
--------------------
Alice,Big Spring,Colorado City,Denton,Eagle Pass,Frenship,Goliad
Frenship
Alice<->Big Spring,Alice<->Colorado City,Big Spring<->Colorado City,Big Spring<->Denton,Big Spring<-
>Eagle Pass,Colorado City<->Denton,Colorado City<->Frenship,Denton<->Eagle Pass,Denton<-
>Frenship,Frenship<->Eagle Pass,Frenship<->Goliad,Eagle Pass<->Goliad
--------------------
Caldwell,Bryan,Navasota,Hempstead,Franklin,Calvert
Bryan
Caldwell<->Bryan,Bryan<->Navasota,Navasota<->Hempstead,Hempstead<->Franklin,Franklin<-
>Calvert,Calvert<->Hempstead,Bryan<->Calvert
--------------------
Texline,Brownsville
Texline
Texline<->Brownsville
--------------------
```

**Sample output:**
```
Test case 1: possible
Test case 2: possible
Test case 3: impossible
Test case 4: possible
Test case 5: impossible
Test case 6: possible
Test case 7: possible
Test case 8: impossible
Test case 9: impossible
```